REMARKS/ARGUMENTS

These remarks are submitted in response to the Office Action of October 6, 2005 (Office Action). As this response is timely filed within the 3-month shortened statutory period, no fee is believed due.

In paragraphs 4-5 of the Office Action, Claim 12 was rejected under 35 U.S.C. § 112, as being indefinite for failing to particularly point out and distinctly claim the invention. In paragraphs 6-7 of the Office Action, Claims 1-13 were rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,345,450 to Saw (hereinafter Saw).

Independent Claims 1, 5-8, and 12-13 have been amended to further emphasize certain features of Applicants' invention. Claim 12 has been amended to correct the insufficient antecedent basis noted in the Office Action. These amendments are supported throughout the Specification, and no new matter has been added by virtue of the amendments.

I. Applicants' Invention

It may be helpful to reiterate certain aspects of Applicants' invention prior to addressing the references cited in the Office Action. The invention, according to one embodiment, provides a method and apparatus for eliminating redundancy among multiple execution sequences during workload simulation on an e-business application. The method can include the steps of creating a workload reference object having one or more reference command objects. In response to a work request to process a workload, selected ones of the reference command objects can be cloned or copied. The cloned or copied reference command objects can be assembled to create the workload required by the work request. The assembled command objects can subsequently be executed.

The step of creating the workload reference object can also include the step of parsing workload configuration data to create a master workload. Since the master

workload contains all the workloads for the system and can be used to generate workloads, it can be appropriately called a workload generator. The workload configuration data that gets parsed can be stored in a file such as a workload configuration file. In response to a request to add a new command, the workload configuration data stored in the workload configuration file can be modified. A workload reference object can be created using the workload configuration data. The created workload reference object can subsequently be assembled to create a workload executable, which is data that can be executed.

Applicants' invention facilitates the development of a workload simulation on an e-business application server by simplifying the process in which commands are presented in a workload. In particular, Applicants' invention provides an administrator the ability to replace redundant command sequences with a workload identifier. The workload identifier can then be used in place of the redundant command sequence. The administrator can modify the workloads being requested using the workload identifier without having to rewrite the redundant command sequences. Applicants' invention includes a workload generator that dynamically generates workloads from a configuration file. The configuration file can be updated by a system administrator to define a workload or set of commands to be executed across a set of e-business systems. The workload configuration file can be a text based file that can be easily modified by the administrator (a non-programmer). The workload generator reads the workload configuration file and enables the execution of the redundant commands sequences in an order specified in the configuration file.

For example, an HTTP request or servlet may request a task for calculating an order total which may require processing multiple redundant commands. In practice, it can be expected that general client requests will contain similar tasks for which an e-business application server is expected to respond. In many cases, it is also essential that systems responsible for processing the task follow the same sequence (i.e. order) in

which commands are presented. A work request can contain multiple commands that should be followed in a particular sequence. Notably, the e-business application server is responsible for coordinating the sequence of these tasks in responding to the work request. The e-business application server delegates the tasks to the systems within the distributed network according to the sequence of commands within the work request through a workload. Applicants' invention can also create a workload reference object which defines the correct order for responding to the work request. The sequencing (i.e. ordering) of the commands described within the workload reference object can be maintained during program execution. Program execution can be regarded as the set of processes associated with responding to the task. For example, there may be a specific sequence of commands associated with a servlet request for calculating the order total.

The e-business application server is responsible for the ordering of the commands. This can be a redundant task when similar work requests or portions of work requests are received. For example, the same resources and configurations are used when responding to commands of a redundant work request. Understandably, a work request that requires systems within the distributed network to coordinate efforts for addressing a similar work request can produce unnecessary processing overhead. In one aspect, Applicants' invention overcomes redundancy by cloning commands in a saved configuration file for later deployment. Each configuration can produce a workload reference object that combines various reference command objects particularly suited for responding to a work request. The work load reference object can be subsequently assembled to create a workload executable that describes how the distributed systems will be called and accessed in responding to a work request.

In Applicants' invention, reference command objects of the workload reference object can be reused instead of regenerating new objects to process redundant tasks. In effect, Applicants' invention screens work requests for redundant commands or portions thereof that are common to other work requests previously identified. A configuration

file is generated to capture the ordering of redundant command sequences. The configuration file identifies the workload reference objects that are capable of responding to the redundant commands without having to allocate resources to rebuild new command objects. The configuration file stores command objects capable of responding to previously similar work requests that can be later accessed for responding to a work request.

In particular, Applicants' invention allows an invoker (e.g. a system administrator) to access the configuration file and create work requests for development testing or for simulating the e-business application server. For example, an administrator can create a suite of work requests to test the performance of the e-business application server without rebuilding code objects. The system administrator can manually edit the configuration file to change the order in which work commands are processed without rebuilding workload reference objects. In another example, a system administrator can modify the manner by which commands of a work request are delegated to application systems within the distributed network without having to rebuild workload reference objects or without having to reprogram the workload generator. For example, during performance monitoring, a system administrator can reallocate responsibilities within the distributed network to balance overload conditions. As previously mentioned, this facilitates the ease by which the workload configuration order of an e-business application server can be handled at the administration level.

II. The Claims Define Over the Prior Art

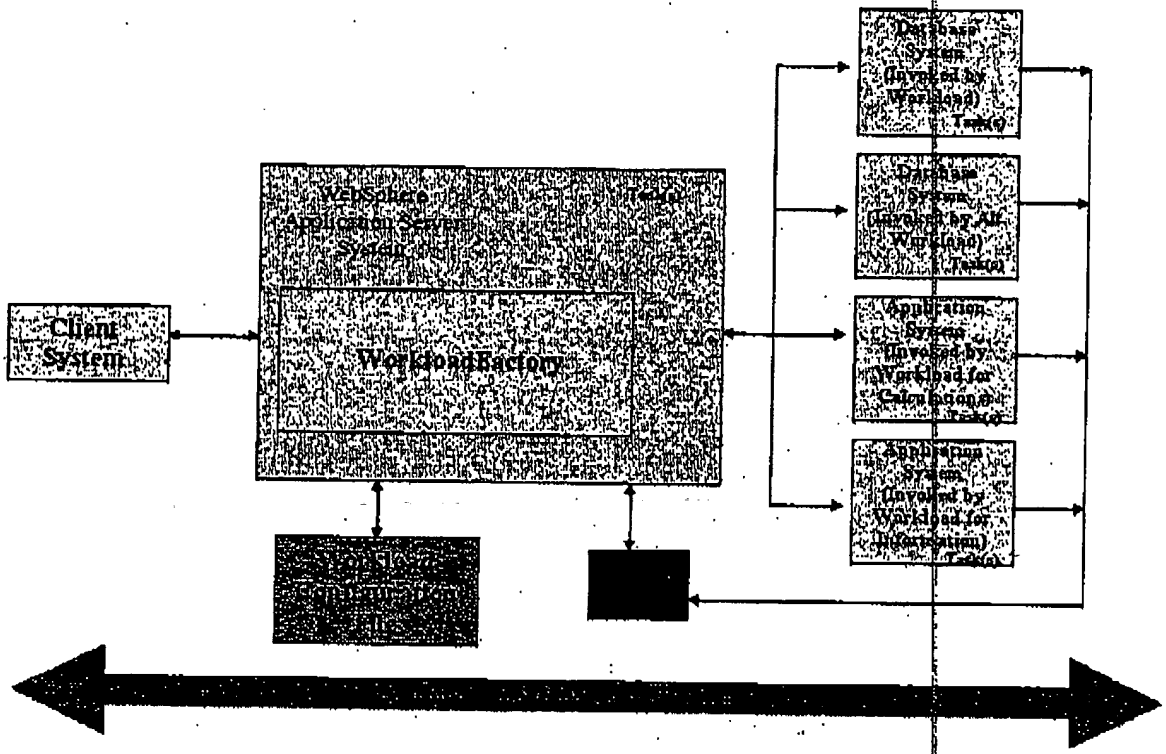
As already noted, independent Claims 1, 5-8, and 12-13 were rejected as being anticipated by Saw. Applicants respectfully submit, however, that Saw fails to expressly or inherently teach every feature of Claims 1, 5-8, and 12-13, as amended.

Saw is directed to a method for simulating a logic device on a single system. The method is appropriate for generating a test program for testing the logic device on a logic

device tester. Saw provides a granular functionality for creating input and output vectors to be applied for logic devices or digital circuitry which responds to electrical signals received through the plurality of input channels. These input channels require a physical connection (electrical voltage) to the device.

In contrast, Applicants' invention provides the concept of an e-business application workload which can span multiple systems and can modify the work or workload being requested of various back-end systems. For illustration, a diagram is presented below to depict the e-business application system and which is composed of multiple systems. The lines in red illustrate the network connections which are used to interconnect the various systems together. The green arrow near the bottom of the diagram depicts that the e-business system is comprised of a network infrastructure, a web application server, and various back-end systems and workloads which span across these various systems. Applicants' invention takes into account the distributed nature of application development wherein an application can be divided into multiple components operating on multiple systems. The workload generator enables the dynamic creation of workloads and commands being performed to support a specific workload across multiple systems (web application server, back-end system, database system, etc.). Applicants' invention provides an administrator the ability to modify the workloads being executed without programmatic changes to the workload generator. The workload generator and "execution sequences" being referred to in the specification provide the flexibility for a system administrator to reconfigure the workloads being performed across a variety of back-end systems.

Components of an e-business system



Claim 1

With regard to Saw (Column 3 lines 26-35), the redundant input vectors being created are put in place as input for logic devices. These input vectors contain a set number of input and output control bits of information. This type of test driver deals specifically with "high" and "low" logic voltage levels. In contrast, Applicants' invention concerns testing of an entire set of e-business systems wherein the workloads being created are not created programmatically, but rather by a system administration (See Applicants' Specification, Pg. 9 of XML configuration file showing a user is a database administrator <User>"db2admin"</User>). The workload configuration file is a text based file that can be easily modified by the administrator (a non-programmer). The workload factory simply reads the workload configuration file and enables the execution

of the commands specified in the configuration file. Applicants' invention also differs in that there is no set number of commands executed. This is dynamically driven by what commands are specified for a workload in the configuration file.

With regard to Saw (Column 2, lines 66-69), the generation of sufficient simulation data to automatically generate a test program from the simulation data is presented. The vector input and output data are specific for the device being tested and the system requires an understanding of this from a programmatic perspective in order to ensure appropriate functionality of the simulator. Applicants' invention differs entirely from this scope. Applicants' invention does not deal with the input and output of the commands being executed in a workload across the set of systems being exercised. Predefined inputs and outputs are not required, nor the setting up of workloads to be done in a programmatic fashion.

With regard to Saw (Column 3, lines 45-49), the duplication of the input vectors is for the purpose of expanding the set of test vectors used to generate the test program. Once again, this approach is programmatic in nature and is done in this fashion since the test program will need to produce electrical signals. Applicants' invention is different in that the workloads are not specific to a particular device or system. The workloads are defined across a variety of systems and provide business level functionality.

With regard to Saw (Column 2, lines 40-45), the tabulated and corresponding values are signal values for the input and output vectors and are "computer generated". Applicants' invention is unique in that the workloads are comprised of business level functionality.

Claim 2

With regard to Saw (Column 3, lines 45-49), the forming of an expanded set of test vectors requires the same number of input and output vector elements. Applicants' invention differs in that the workloads created perform functions across a multitude of

different systems. Once again, Applicants' invention does not require programmatic changes in order to change the workloads being described, but rather a simple text change by a business system administrator.

Claim 3

With regard to Saw (Column 4, lines 60-65), Applicants' invention differs from Saw in that workloads contain commands which are not specific to a particular logic device. The workloads and commands that Applicants' driver can execute provide business level function as opposed to logical device function.

Claim 4

With regard to Saw (Column 7, lines 38-45; Column 2, lines 66-69; and, Column 2, lines 40-45), Saw programmatically increases the number of input and output vectors being utilized for testing. Saw applies a sequence of input vectors with a predetermined set of output vectors for testing ROM. Applicants' invention involves a set of e-business systems that does not require specific knowledge of the output of the workload commands being executed. Applicants' invention deals with workloads which execute business level function as opposed to logical device function.

Claims 5, 6, and 7

With regard to Saw, the simulated data refers to circuit data with "high" and "low" values. In contrast, workloads and commands of Applicants' invention are business level functions that are not specific to a particular device.

Claims 8, 9, 10

Saw does not create a flexible driver for an e-business application system. In addition, the techniques described by Saw could not be applied to drive an e-business

system. Applicants' invention is unique in that it provides a mechanism to create dynamic workloads and commands across a multitude of e-business systems.

Applicants respectfully assert that whereas Saw fails to expressly or inherently teach each of the features recited in amended independent. Moreover, the prior art fails to provide any suggestion or motivation for extending Saw's device test program to an e-business application system. Therefore, independent Claims 1, 5-8, and 12-13 are not anticipated by the prior art. Applicants further respectfully assert that whereas each of the remaining claims depend from one of amended independent Claims 1, 5-8, and 12-13 while reciting additional features, these dependent claims likewise define over the prior art. Applicants, therefore, respectfully request withdrawal of the rejection of Claims 1-13.

CONCLUSION

Applicants believe that this application is now in full condition for allowance, which action is respectfully requested. Applicants request that the Examiner call the undersigned if clarification is needed on any matter within this Amendment, or if the Examiner believes a telephone interview would expedite the prosecution of the subject application to completion.

Respectfully submitted,

Date: January 6, 2006

Gregory A. Nelson, Registration No. 30,577
Richard A. Hinson, Registration No. 47,652
Marc A. Boillot, Registration No. 56,164
AKERMAN SENTERFITT
Customer No. 40987
Post Office Box 3188
West Palm Beach, FL 33402-3188
Telephone: (561) 653-5000